NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

DIGITAL AUTHENTICATION FOR OFFICIAL BULK EMAIL

by

Andrew A. Slack

March 2009

Thesis Advisor:	Simson L. Garfinkel
Second Reader:	Geoffrey Xie

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** March 2009	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE Digital Authentication for Official Bulk Email		**5. FUNDING NUMBERS**	
6. AUTHOR(S) LT Andrew A. Slack, USN			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000		**8. PERFORMING ORGANIZATION REPORT NUMBER**	
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A		**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**	
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited		**12b. DISTRIBUTION CODE**	

13. ABSTRACT (maximum 200 words)

Official bulk email is an efficient tool for disseminating information to a wide audience. Its inherent efficiency, captive audience, and trust provide a dangerous attack vector for adversaries utilizing fraudulent email.

Digital authentication can provide a layer of defense to official bulk email that, combined with other defensive countermeasures, will greatly reduce its vulnerabilities. The Department of Defense mandates that official emails, which contain hyperlinks, attachments, or instructions to recipients, must contain a digital signature, authenticating the source of the email, and ensuring the integrity of its contents. This policy, though used at some military installations, is not being applied to official bulk email at others due to administrative roadblocks in obtaining role-based certificates, and implementing an authentication policy with legacy email systems.

This thesis identified administrative roadblocks in deploying digital authentication solutions within the Department of Defense, explored different technology options of a digital authentication solution for official bulk email, created a proof of concept solution using a Python proxy server and S/MIME, and looked at the most popular mail user agents to see how they interpret S/MIME digital signatures. Applying digital authentication to official bulk email will close a potentially critical vulnerability in the defense of DoD networks.

14. SUBJECT TERMS Digital Authentication, S/MIME, Official bulk email, phishing			**15. NUMBER OF PAGES** 77
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited

DIGITAL AUTHENTICATION FOR OFFICAL BULK EMAIL

Andrew A. Slack
Lieutenant, United States Navy
B.S., United States Naval Academy, 2002

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
March 2009

Author: Andrew A. Slack

Approved by: Simson L. Garfinkel
Thesis Advisor

Geoffrey Xie
Second Reader

Peter Denning
Chair, Department of Computer Science

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Official bulk email is an efficient tool for disseminating information to a wide audience. Its inherent efficiency, captive audience, and trust provide a dangerous attack vector for adversaries utilizing fraudulent email.

Digital authentication can provide a layer of defense to official bulk email that, combined with other defensive countermeasures, will greatly reduce its vulnerabilities. The Department of Defense mandates that official emails, which contain hyperlinks, attachments, or instructions to recipients, must contain a digital signature, authenticating the source of the email, and ensuring the integrity of its contents. This policy, though used at some military installations, is not being applied to official bulk email at others due to administrative roadblocks in obtaining role-based certificates, and implementing an authentication policy with legacy email systems.

This thesis identified administrative roadblocks in deploying digital authentication solutions within the Department of Defense, explored different technology options of a digital authentication solution for official bulk email, created a proof of concept solution using a Python proxy server and S/MIME, and looked at the most popular mail user agents to see how they interpret S/MIME digital signatures. Applying digital authentication to official bulk email will close a potentially critical vulnerability in the defense of DoD networks.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank my friends who helped me through life while working on my thesis: Paul Farrell and our time spent at Trailside, Deane Smith, John Kajmowicz, Todd Glidden, Jessy Cowan-Sharp, and Pete Corrao for his leadership and guidance.

I would also like to thank the members of the NPS Information Technology and Communication Services (ITACS) for their help in making me understand the NPS infrastructure, letting me see the certificate acquisition process first hand, and reviewing the thesis.

My Mother, Christa, my Father, Jim, and my Brother, Jimmy, were always supportive and kept me focused on graduating. They were always there to lend an ear or a bit of advice.

Finally, I would like to thank my thesis second reader and advisor. Professor Geoffrey Xie allowed me to switch thesis topics after I had started, and provided support as I researched and learned about digital authentication and official bulk email. Without Professor Simson Garfinkel as my thesis advisor, I would have never finished my thesis on time. His vast experience, infallible work ethic, and compassion for his thesis students did not go unnoticed by my friends, my family or myself. Thank you Dr. Garfinkel.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. OFFICIAL BULK EMAIL

Official bulk email is a common way for administrative managers to get information out to the workforce. It allows the sender to push a single button to get announcements, tasking, or other administrative direction to everyone within their domain or sub-domain.

Most of us are accustomed to receiving official bulk email, and we typically accept its contents as valid without checking integrity of the message or the authenticity of the sender. This inherent trust of official email provides a dangerous attack vector to people who want to steal information, pass misinformation, or install malicious software. This type of attack, where email is used to convince people to convey personal information, is called Phishing [1].

B. PHISHING

Phishing attacks against specific institutions, such as Department of Defense (DoD) commands, are becoming more common. The attacks are also becoming more sophisticated, spoofing actual official email sources with specifically crafted content for the target recipients. The results of these attacks range from compromised user accounts to the compromise of personal information and services. As individuals and organizations improve their attack techniques, more and more accounts and critical information will be compromised [2].

Currently our main defense against these crafted emails is filtering. Firewalls and spam blockers do a good job in keeping out known vectors of attack such as blacklisted domains or executable file attachments, yet they are only effective against attacks that can be identified. This is an AI problem, as the phishing defense must make a decision to allow or deny access to the network for each email it sees based on the email's content. Attackers understand this, and can eventually defeat any defense that relies on content-based filtering.

C. CRYPTOGRAPHIC AUTHENTICATION AS DEFENSE

Cryptographic authentication can severely limit the success rate of spoofed email, since it is computationally infeasible to defeat modern signature algorithms. By using cryptographic authentication in conjunction with current defenses, like filtering, spoofed emails may be prevented from entering the network altogether. In the future, anti-spam software may automatically verify digital signatures on incoming email messages, removing the burden from the user. The Department of Defense has already deployed software that will verify digital signatures on the desktop, and has even created policy that requires a digital signature for emails that contain specific types of data [3]. Yet this policy is not widely enforced and many users don't understand the need for digital authentication, or how to use it properly. Furthermore, many administrative roadblocks exist that delay the deployment of digital authentication technology.

D. PURPOSE OF STUDY

This thesis analyzes the problems associated with deploying a digital authentication solution to auto-generated email, researched different commercial solutions, and present a proof-of-concept script that can sign auto-generated emails through the use of a proxy server. It reviews common Mail User Agents (MUAs), shows how they display S/MIME signed messages, and characterizes a bug in how Apple Mail verifies S/MIME signatures specifically with Department of Defense Common Access Cards (CAC).

We focused on automated email versus interactive email for a number of reasons:

1. Common Access Cards (CAC) that have been deployed by DoD will apply a personal digital signature to interactive email, but cannot be used with automatically generated email or with role-based certificates.
2. Automated emails usually have no reply-requirement, bypassing the usability problem associated with requiring a digital signature for replies.
3. Finally, because automated emails prime users to accept unsigned fraudulent commands, and currently, no one else is attempting to apply a digital signature to them.

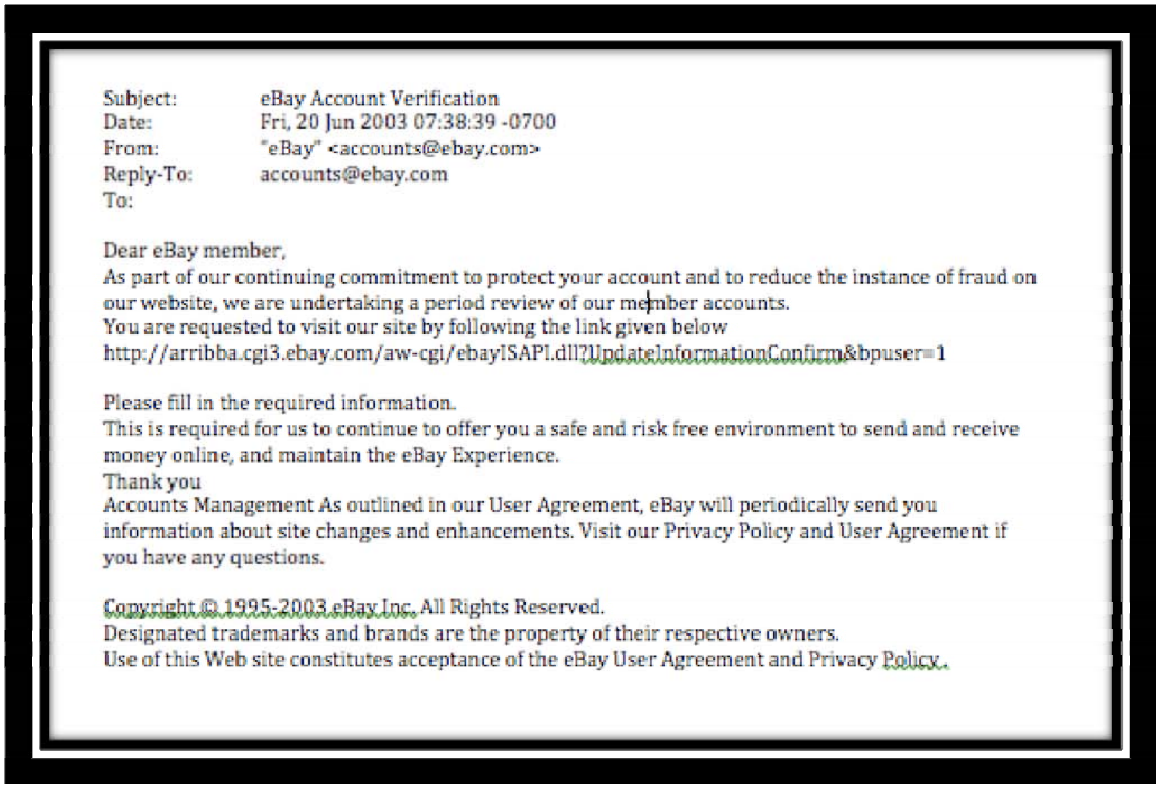
THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND AND RELATED WORK

A. THE THREAT – OFFICIAL BULK EMAIL AND PHISHING

Official bulk email provides a means to disseminate official information to an entire organization with ease. These emails can be fairly common, and are mostly automatically generated. Because of their frequency and official nature, auto-generated bulk email evoke an inherent trust response with their recipients. Attackers can easily exploit both the selected distribution of official bulk email and their perceived integrity.

This use of fraudulent email is called “phishing,” a term coined in the early days of computer hacking. These fraudulent email attacks have become more technical, more personal, and more successful each year. Criminals have devised ways to increase the success rate of their attacks, including targeting specific users with a specific relationship, such as customers of a bank or commerce-relate website. These fraudulent emails look like official communication from the target bank, but will usually contain a malicious attachment, or a link to a website that will try to collect personal information from the victim (see Figure 1) [2].



Subject: eBay Account Verification
Date: Fri, 20 Jun 2003 07:38:39 -0700
From: "eBay" <accounts@ebay.com>
Reply-To: accounts@ebay.com
To:

Dear eBay member,
As part of our continuing commitment to protect your account and to reduce the instance of fraud on our website, we are undertaking a period review of our member accounts.
You are requested to visit our site by following the link given below
http://arribba.cgi3.ebay.com/aw-cgi/ebayISAPI.dll?UpdateInformationConfirm&bpuser=1

Please fill in the required information.
This is required for us to continue to offer you a safe and risk free environment to send and receive money online, and maintain the eBay Experience.
Thank you
Accounts Management As outlined in our User Agreement, eBay will periodically send you information about site changes and enhancements. Visit our Privacy Policy and User Agreement if you have any questions.

Copyright © 1995-2003 eBay Inc. All Rights Reserved.
Designated trademarks and brands are the property of their respective owners.
Use of this Web site constitutes acceptance of the eBay User Agreement and Privacy Policy.

Figure 1. Fraudulent Email Example (From 4)

B. SOCIETAL ASPECTS OF PHISHING ATTACKS

Phishing attacks also have indirect consequences as well. Societal aspects, such as trust, support, and reliable communication, are victims of phishing attacks in addition to direct monetary loss [5]. Even if a phishing email is unsuccessful, the ensuing trust in whatever company or organization that attack mimicked is degraded. For example, a phishing attack using an address that looks like the Red Cross may make actual Red Cross emails requesting aid for the current crisis less effective. This will have a negative impact on legitimate organizations' ability to raise funds through email campaigns [5]. A similar outcome occurs with banks that are imitated by phishing scam

artists. Users will be more afraid to follow links enclosed with legitimate bank emails. Banks will have to either follow safer email practices, such as using digital cryptographic authentication, or fall back on alternate methods of communication which may be more expensive or wasteful (e.g., phone calls and paper mail).

One company, PayPal, is attempting to initiate a fundamental change in how email providers handle their mail. PayPal's general council states that their company is one of the highest targeted e-commerce companies currently in business by phishing attacks. PayPal sends a significant amount of mail to its customers, and criminals have targeted PayPal and those customers by sending spoofed messages that appear to come from PayPal but direct customer to fake websites. As a result, in 2007 PayPal asked all major email service providers, such as Google and Yahoo!, to block any email that claims to comes from the PayPal.com domain, but do not contain PayPal's digital signature [6].

PayPal was already combating phishing attacks against their customers by limiting the amount of money per transaction and compensating users who where victims of phishing attacks. By attaching a digital signature to their official mail, they are helping both email service providers identify fraudulent mail, and adding another layer of protection for their customers.

C. MITIGATING PHISHING BY TRAINING

One of the weakest links in computer security is the user. Nevertheless, training of users has traditionally been a lower priority among security professionals than

preparation, avoidance, intervention, and treatment [7]. Because of this, the effects of user training are mixed. When users are confronted with traditional context-free phishing attacks, training has shown to significantly reduce the number of users who fall victim. Yet when trained users are presented with sophisticated context-specific attacks (spear-phishing), training seems to have reduced impact. Training also increases the likelihood that legitimate emails will be labeled as attacks. This, researchers state, is because users, despite training, cannot recognize what phishing is, how to identify phishing attacks, if they understand what it is, or have an inherent trust that attacks shouldn't come from recognized sources. This is a troubling conclusion for organizations that send bulk email, as it implies that training may make bulk email less effective and that even with training, these organizations will remain susceptible to targeted phishing attack.

Training is a base requirement to mitigate and prevent phishing attacks, but as these attacks become more context-sensitive, training will become less effective by itself. Users need both training and extra tools to either assist in training or to assist in identifying phishing attacks.

Because spear-phishing uses phrases, terminology, and target a select group of people, automated systems have a harder time identifying and filtering out these attacks. Context specific emails will get through most generic filters, because they are intended for a smaller audience, and have specific details relating to current or recent projects. If automated systems had a high enough threshold to weed out these context-specific attacks, then there is a

high probability that the filters would prevent the delivery of legitimate mail as well.

Context-specific fraudulent emails are increasing every year in both number of attacks and sophistication because they are successful. By targeting specific users, the criminal can bypass most automated security systems and increase the chances of a successful attack. The more convincing their fake emails look, the more untrained users are likely to fall victim. Even users who were given basic training to identify fraudulent email still fell victim to the more sophisticated and authentic looking ones.

D. THE WEST POINT EXPERIMENT

In 2004, researcher Aaron J. Ferguson conducted a phishing experiment at West Point Military Academy [8]. He sent out a specially constructed email with an embedded hyperlink to a group of 512 randomly selected cadets. This target group consisted of an even number of freshmen, sophomores, juniors, and seniors. Ferguson selected West Point because it is a DoD facility, has a Computer Emergency Response Team (CERT), and was the only service academy at the time to be certified by the National Security Agency (NSA) as a Center of Academic Excellence in Information Assurance Education (CAEIAE). A final important factor was the fact that each freshman class underwent four hours of information assurance training, where they discuss various security practices related to electronic communication, including phishing.

Ferguson's phishing email was crafted to exploit the military mentality of the cadets, yet have obvious errors that should have caused suspicion among the recipients. The email was sent from a Colonel, a high-ranking officer, and asked the students to follow an included hyperlink, but the Colonel was not in West Point's global address book. The Colonel also listed his office on the 7th floor of a well-known building on campus. The cadets should also have known there was no 7th floor of this building. The experiment resulted in 80% of the cadets of different rank clicking on the hyperlink and 90% of the newly trained freshman falling victim. Ferguson showed that even a select group of individuals with specialized training and within an organization that has a greater need of cyber security can still fall victim to specially crafted spear-phishing attacks.

E. DOD TARGETED PHISHING

Phishing campaigns have expanded from ISP users and banks to governmental organizations like the Department of Defense. These more sophisticated attacks, with emails that look identical to official mail, are the most threatening to the security of Government networks and DoD members [2].

The United States remains the largest host of phishing websites in the world [9]. This is not an indication of malfeasance on the part of its citizenry, but a result of bandwidth, target populace, and privacy laws. The US has both more aggregate bandwidth than any other country in the world and the largest number of unmanaged home computers – each a potential launching pad for attack. America is also the world financial leader, so the same users who do their

banking online are the primary targets of financially focused phishing attacks. Finally, the US has strict laws protecting the personal rights of its citizens, which help the phishing criminals since authorities must pass through multiple legal obstacles to seize and analyze computers used in phishing attacks.

Phishing attacks within the DoD are not necessarily attempting to gain financial information. They may be targeted at government members to gain intelligence or account information [2]. From these compromised accounts, further exploitation of DoD network may be possible. In a training presentation, released by JTF-GNO in late 2006, the DoD identified the sophistication of adversaries and their techniques [10]. They call these focused attacks via email "spear-phishing." Many of these malicious emails identify the intended victim by name, contain attachments relevant to ongoing exercises, and use jargon associated with the false intent of the email. This leads investigators to believe that some attackers already have extensive knowledge of their targets, and know precisely what further information they want.

F. EMAIL HISTORY (SMTP, SECURITY)

Early email systems could only transmit text messages. As a result, the first email standards only specified how to construct the message headers (From, To, Date, and Subject) [11]. These standards were silent on the topic of security. For example, RFC-821 defines the SMTP protocol, but does not even mention the words "security," "authentication," or "encryption" [11].

Email was originally based on a protocol that does not inherently authenticate people, or provide for secure communications. The sender was identified by the From: address, but not authenticated. Although it was recognized that the From: address could easily be forged, the designers did not have the cryptographic tools available to allow authentication in a distributed environment. Besides, at the time email was primarily used as a way for researchers from different institutions to communicate – there was no credible threat requiring email to be protected. This lack of baseline security slowed the progress of secure SMTP as a standard and enabled criminals to use email as a primary attack tool.

As attacks and spam became more prevalent, a new RFC was created, RFC-2821, that now recognized these security concerns, and suggested "end-to-end" methods are the only real security solution [12]. Unfortunately the RFC goes on to state, "This specification does not further address the authentication issues associated with SMTP…." In 2008, the SMTP RFC was updated again in RFC-5321, but no further security extensions were proposed, only an expanded discussion of security vulnerabilities [13].

G. SECURE DIGITAL MESSAGING

People have realized the importance of secure digital messaging since the email user base began to grow from the small group of researchers to the wider public. Encryption schemes have been in use for conventional messaging throughout history, but there was always the problem of transporting the keys or the secret way to decrypt messages securely. This need drove the development of public key

cryptography. Algorithms such as Diffie-Hellman and RSA were developed specifically for the purpose of securing electronic mail - both by adding privacy (encryption) and by providing authentication of the sender.

H. USABILITY OF SIGNED EMAIL

Many researchers have blamed the lack of secure email deployment not on the competition or technical shortcomings of the various proposals, but on fundamental usability problems resulting either from poorly designed software, overly complex protocols, or a mismatch between the security requirements of PKI and the real-world needs of organizations.

It has been shown that despite the strength of the cryptography or the global acceptance of a protocol, if a user has trouble sending a secure message, the goal of email security has failed [14]. Whitten famously conducted a study of 12 subjects, who were given the task to create public/private key pairs, then send an encrypted and signed email with PGP 5.0; only a third were successful. Whitten's experiment showed that even with training, many people have difficulty using security software that requires significant user participation. Usability is one of the major roadblocks to the adoption of digital signature software, and is why the technology is so slow to reach mainstream adoption.

While some researchers feel that it is important to teach normal users exactly how digital cryptographic algorithms work, others argue that regular users do not need such in-depth knowledge [15]. A basic understanding of what

digital encryption and authentication provide is enough to make the technology useful.

As the number of users on the Internet grew, so did the problem of spam and phishing. This increasing threat eclipsed the usability problems associated with key pair generation and encryption for an immediate need of simple mail authentication with signatures. Some researchers argue that solving the problem of encryption must wait until the phishing and spam threat are mitigated. But digital authentication protocols are at a state where most users can receive digitally signed messages because most email programs already have cryptologic technologies built in [16]. Nevertheless, despite the widespread deployment of this technology, very few emails are sent digitally signed. Again, usability in conjunction with the users' perceived indifference toward secure messaging is suggested as the reason [16].

Another roadblock for the adoption of secure email (specifically S/MIME) is the required use of authenticated certificates. It is a burdensome task to obtain a certificate from a reputable certificate authority (CA), and the use of self-signed certificates pops up an alert on many mail user agents [16].

Despite this pop up, Garfinkel's 2005 study of the usability of signed email found that mail signatures, rather than email encryption, could effectively help users withstand a laboratory phishing attack; subjects were protected from the attack even when using self-signed certificates when provided with software that implemented the Key Continuity Model [17].

Digital signature technologies are deployed on most user email clients today, and do not require recipients to obtain their own certificate to verify signed messages. Though usability of these programs is still a large obstacle to overcome, institutions are primed and ready to implement signing rules to a wide spectrum of their email, and can bypass much of the usability issue by incrementally employing the authentication protocols to certain types of email, like automated messages.

I. THE IMPORTANCE OF SIGNED EMAIL

Other work has found that even when information workers appreciate the importance of encrypting their mail, they generally do not understand the advantage to signing their mail. Gaw, Felten and Fernandez-Kelly conducted a series of interviews at ActivistCorp, a non-violent, direct action organization: "Although we had not explored the topic in depth, digital signatures seemed relatively unimportant to the employees we interviewed" [18]. The employees they interviewed at ActivistCorp stated that they had reason to maintain the secrecy and integrity of their messages. Most users knew how to encrypt, and assurances encryption provides, but few understood the use or importance of digital signatures, and would only sign their message if it was encrypted [18].

Fritsche and Rodgers evaluated a range of cryptographic technologies for deployment at Lehigh University. They considered a range of technologies in including hardware encryption, secure messaging, and network security. They made a comprehensive list of recommendations but only mentioned email signatures at the end of the article. They

focused on solutions that require extensive work or a significant change to the way the school communicates, and did not discuss the protocols for simple email authentication [19].

J. PEM

Work on the Privacy Enhanced Mail (PEM) standard began in the mid 1980's. PEM provided end-to-end security for email based on public key cryptography. The PEM design was finalized in 1993 [20]. PEM provides for both message sealing and signing. It seals the message by encrypting message contents with a symmetric encryption algorithm, and then encrypts the session key with the recipient's public key. PEM signs the message by creating a digital hash of the message, and encrypting that hash with the sender's private key. PEM was designed when there was no common repository of authentic public keys, so it used a chain of certificate authorities to verify the authenticity of the individual public keys, based around the trust of a single root server [20]. This centralized root server became problematic as people discovered the costs and legal ramifications of a trusted hierarchical structure.

K. PGP

Pretty Good Privacy is a program first developed and released by Phil Zimmermann in 1991 [21]. PGP did not use a centralized trusted root server for a chain of trust; Instead, PGP enabled users to trust whomever they wished. The idea was that untrustworthy certificates (and the organizations or people who sign them) would fall to the wayside as more trustworthy organizations rose to the top,

creating a “web of trust.” An early example of a community reputation model - PGP was not easily applied to current email programs since it required a lot of configuring and user knowledge. Though usability was significantly improved in 1997 when PGP was commercialized, PGP was never widely used outside a select group of cryptography advocates and human right activists.

L. S/MIME

Development work on the Secure Multipurpose Internet Mail Extensions (S/MIME) began soon after PEM was standardized. Users and developers were discovering the problems associated with the hierarchical chain of trust ending in a single root server, so S/MIME relaxed this policy. Its developers envisioned a network of trusted certificate authorities, any of whom could provide the trust for individual certificates. Even more convenient, the software makers implementing S/MIME could include these pre-determined and trusted CA public keys with their software distribution.

In 1996, Microsoft Corporation announced it would be including S/MIME in their mail service products (Outlook, Exchange client, and Internet Mail). This sparked Microsoft’s competitor, Netscape, to also include S/MIME support in its products. Early support for S/MIME from these industry giants pushed the secure messaging protocol into the homes of millions of users (most without realizing they had the functionality). Today, most commercial email mail user agents (MUAs) support S/MIME. This wide-spread support is one of the greatest advantages for S/MIME.

S/MIME is not used by a majority of individuals though. Many webmail systems do not support S/MIME, although there is support for S/MIME signatures in Outlook Web Access, and some support for Gmail with a browser plugin. S/MIME also suffers from the same problems that trouble public key infrastructure. There are also some ambiguities in the RFC describing certificate format, which may lead to incompatible S/MIME implementations. Users must also first obtain a personal certificate before they can digitally sign or encrypt email. This process is intimidating and can be confusing to people who do not fully understand the method of acquiring a valid personal certificate. Finally, S/MIME authenticated messages contain a multipart section with a .p7s file extension for the digital signature. Since all mail user agents do not implement S/MIME, individuals may get confused when this signature is displayed as an attachment.

M. DKIM

DKIM is yet another attempt to create an end-to-end authentication scheme for digital mail. Its goal is to overcome the problems with S/MIME. DKIM was started as DomainKeys, a system developed by Mark Delany at Yahoo! . Yahoo! and Google deployed DomainKeys in 2004 on a trial basis to mutually verify mail leaving and entering their respective domains. At the same time, Cisco Systems was developing its own email authentication option, called Internet Identified Mail (IIM). Cisco and Yahoo! began working on a new standard that combined both DomainKeys and IIM; This resulted in a formal IETF proposed internet standard in RFC 4871 as DomainKeys Identified Mail (DKIM).

DKIM signs mail at the domain level. That is, a message originating from a user (e.g. user@nps.edu) is signed by the domain (nps.edu) and not with the user's personal certificate. Whereas S/MIME adds an attachment to a message, DKIM adds new fields in the message header; these fields are simply ignored by mail servers that do not support the DKIM standard. This advantage allows DKIM to be deployed incrementally since it has no impact on users whose software does not support the standard, unlike S/MIME or OpenPGP, where program implementations typically highlight errors rather than ignoring them.

Keys are not assured through certificates (and the signing certificate authority), but by the domain owners themselves. This enables a much more streamlined method of generating a public/private key pair, but also removes a level of trust that is inherent with certificate authorities.

Finally, DKIM relies on DNS. The public key for the domain is stored on the domain's name server, enabling anyone to obtain its public key to verify a signature. This form of key distribution relies on the authenticity and availability of DNS, which itself isn't secure or without error, but has proven itself to be reasonably reliable over the years. DNSSEC is also being deployed, which will prevent spoofing DNS replies.

N. ONGOING DEPLOYMENT ISSUES

Some researchers have suggested that the difficulties in deploying signed email on the Internet today are a result

of the poor match between the S/MIME and PGP protocols and the real-world needs of large-scale organizations.

Goodman, Cormack and Heckerman argue that S/MIME and PGP have not been adopted because of the burdens that the protocols place on the end user and suggest that DKIM will be more successful because it relies on identity at the domain level [22]. They state that user-level authentication is confusing to some users, and require an extra attachment of some form in the email to work properly, where DKIM and SenderID sit at the domain level, relieving the burden on the user [22]. By using domain level authentication, public keys are stored on the domain's DNS server, and the authentication process takes place between servers. The entire process is completely transparent to the user.

Boosting this argument is the fact that major web mail companies like Google and Yahoo! are using this technology today. Yet spammers were also early adopters of domain-verified mail. When SenderID started out, spammers were the ones who were verifying that their spam email came from their spam websites! Domain-level verification works well to prevent an important domain from being spoofed (like PayPal.com), but it doesn't make any guarantees about the nature of the verified mail.

There remain ambiguities and discrepancies with more mainstream protocols like S/MIME. RFC 3850 describes how end-user certificates should be created, to include where email addresses should be place and how verification should check. The problem is the use of the word "should" rather than "must." Because of the suggestive nature of the RFC,

different entities create end user certificates differently, and mail user agents verify them differently. This thesis discovered an example of that problem when the Apple Mail user agent would invariably fail to verify DoD certificates. That issue is discussed in more detail later, but serves as an example of how the S/MIME protocol can be interpreted differently, leading to incompatibility among certificate authorities and verification software.

THIS PAGE INTENTIONALLY LEFT BLANK

III. DIGITAL AUTHENTICATION AND THE DEPARTMENT OF DEFENSE

The Department of Defense understands the importance of high integrity secure messaging. The Joint Task Force – Global Network Operations (JTF-GNO) is a subset of the DoD whose mission statement was developed to ensure, among other tasks, "assured information protection and assured information delivery" [23]. Policies derived emulating from JTF-GNO drive the actions and policy of the rest of the DoD warfighting branches.

A. POLICY

Because of the increased threat to DoD networks by spear phishing, message authenticity and security has become a major component of JTF-GNO's concept of operations. In September of 2008, the Navy issued a restatement of digital signature policy based on JTF-GNO Communication Tasking Orders (CTOs) and prior DoD PKI policy [24]. NAVADMIN 248/08, "Implementation of Navy Electronic Mail (Email) Digital Signature Policy," contains policy for "all unclassified email sent from a Department of Defense (DOD)-owned, operated, or controlled system or account…[for] all emails requiring data integrity, message authenticity, and/or nonrepudiation…" [24]. The Policy states the requirement to apply a digital signature to any email that:

- Directs, tasks, or passes direction or tasking.
- Requests or responds to requests for resources.

- Promulgates organization, position, or information external to the organization (division, department, or command).
- Discusses any operational matter.
- Discusses contract information, financial, or funding matter.
- Discusses personnel management matters.
- The need exists to ensure that the email originator is the actual author.
- The need exists to ensure that the email has not been tampered with in transit.
- Is sent from a DoD-owned system or account which contain an embedded hyperlink (e.g., active link to a web page, web portal, etc.)…
- Is sent from a DoD-owned system or account that contains an attachment (any type of attached file).

The policy also states, “Pure text references (non-active internet links) to web addresses, uniform resource locators (URL), or email addresses do not require a digital signature” [24].

This policy applies both to email from individuals and from email from group accounts, such as automated bulk email.

To help accomplish this requirement, the Navy has issued both Common Access Cards (CAC) and CAC readers to all commands [3][24][25].

B. DOD CAC CERTIFICATES

While working on this thesis, we discovered an inconsistency between the way the Department of Defense creates personal certificates for Common Access Cards and the way that certificates from other sources (such as Thawte and Verisign) are formatted. This inconsistency, combined with an implementation error present in some mail user agents, prevents CAC-signed mail from being properly validated in some cases.

Mail User Agents may verify S/MIME signatures incorrectly. For example, Microsoft products (Outlook, Entourage, etc.) will verify an email address in the "RFC 822 Name" field. This is only place in DoD certificates where email addresses reside.

Other MUAs, including Apple Mail version 3.5 (930.3), will check for an email address appended to the end of the Common Name field. This was a non-standard usage adopted by venders in the early days of S/MIME.

Verisign and Thawte put the email address in both the RFC 822 Name field and append it to the CN field, and Subject Alt Name of x.509 v3 certificates field as follows:

```
Certificate:
      Data:
      Version: 3 (0x2)
      Serial Number: 7d:7e:6d:8e:8c:07:97:f5:f9:58:d0:46:54:c2:ff:94
      Signature Algorithm: sha1WithRSAEncryption
      Issuer: C=ZA, O=Thawte Consulting (Pty) Ltd.,
      CN=Thawte Personal Freemail Issuing CA
      Validity
            Not Before: Dec  8 07:52:47 2007 GMT
            Not After : Dec  7 07:52:47 2008 GMT
      Subject: CN=Thawte Freemail Member/emailAddress=slgarfin@nps.edu
      X509v3 extensions:
            X509v3 Subject Alternative Name: email:slgarfin@nps.edu
      X509v3 Basic Constraints: critical
      CA:FALSE
```

But the Department of Defense will only put email addresses in the RFC 822 Name field (Subject Alternative Name) as follows:

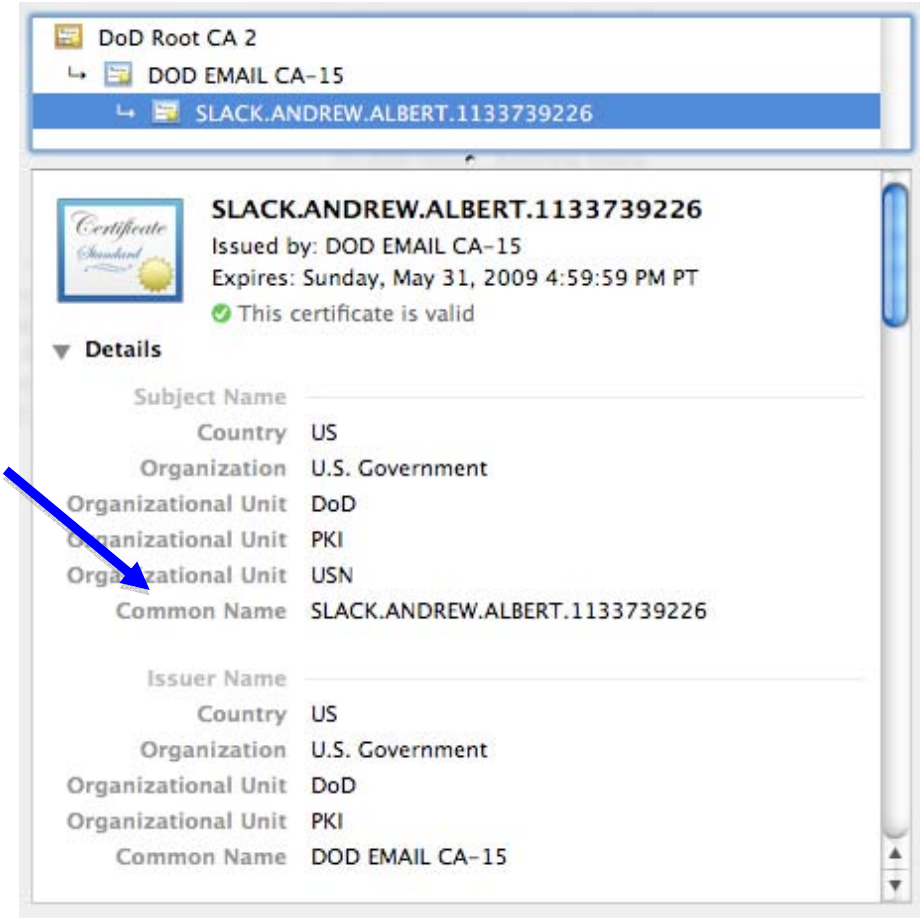


Figure 2. DoD CAC Certificate

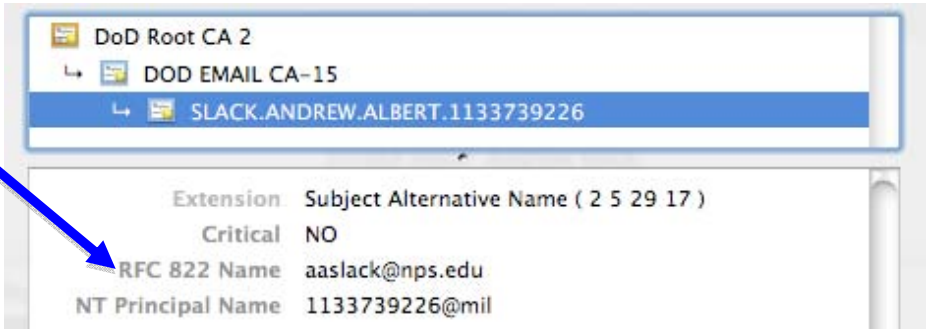


Figure 3. DOD Email Certificate (RFC 822 Name)

This difference will produce the following results in Apple Mail:

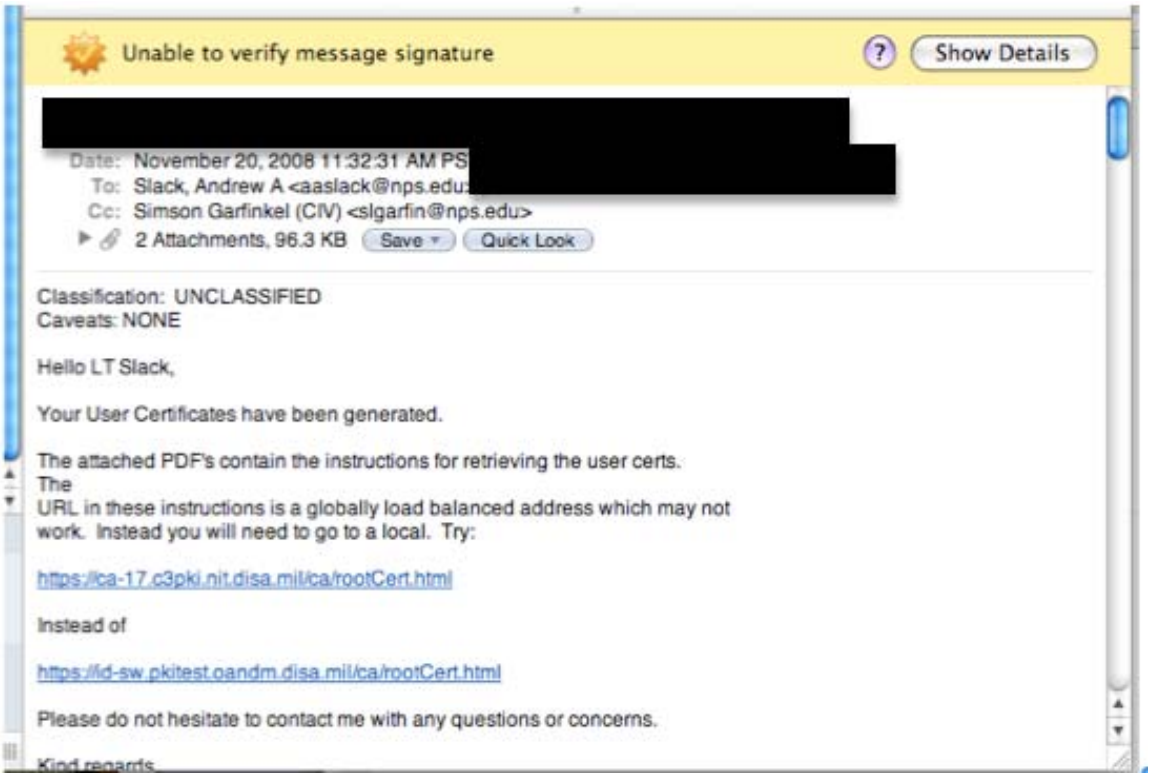


Figure 4. Apple Mail Digital Signature Error

Because the DoD does not put the sender's email address in the certificate "Common Name" field, and Apple Mail didn't check for an email address in the "RFC 822 Name" field, the MUA alerted the email recipient that the signature cannot be verified.

The Apple Mail MUA was verifying S/MIME signatures differently than Microsoft products. A bug report was submitted to Apple. Apple appears to have fixed this bug in OS X 10.5.6 as a result of our bug report. It was an important security concern for the DoD since valid

signatures will be flagged in Apple Mail, undermining the purpose of the digital authentication.

This example is an illustration of how different vendors implement S/MIME. Were S/MIME in wide use today, these problems would have been long ago identified and corrected. It also shows the Department of Defense can work with vendors to have such problems resolved.

Inconsistencies with S/MIME implementation, especially within the Department of Defense, can cause a verification failure. Because all DoD certificates fail to verify in Apple Mail, users of this MUA may begin to ignore all the warnings associated with digital authentication, diminishing the benefits that S/MIME can offer.

IV. NPS EMAIL ARCHITECTURE

The Naval Postgraduate School provides robust email capability to all NPS students, faculty, and associates (contractors, etc.). The system utilizes Barracuda SPAM filters, and Microsoft Exchange 2003 email servers. This chapter evaluates the NPS email system as a case study for how an organization could deploy digital signatures for automatically generated bulk email.

Automated mail at NPS gets generated in a number of ways. One way is from Bulkmail@nps.edu. This mail is created by an authorized user and sent from a program running on an internal server. Other automated mail is generated by SQL@nps.edu, the role-based user for the NPS academic management system. This system regularly sends out reminders to instructors about required actions within the course management system, and can send email to students based on the classes for which they are registered. Mail that is generated internally is sent directly to the exchange cluster, without being processed by the barracuda filters.

NPS hosts email for students, faculty, and associates. DoD regulations prohibit forwarding email outside of NPS.

A. SIGNING OPTIONS

There are several places where NPS could sign automatically generated messages:

a. Message generation
b. After generation
c. Upon receipt

1. Message Generation

The first solution, implement digital authentication at message generation, is the most straightforward solution, but requires modifying legacy code. Also, our target automated systems also are two separate entities, so different scripts have to be modified to accomplish the same task.

This is the most sensible way to sign official bulk email. The required certificates would be located at the message generation servers, matching the security of the scripts to that of the certificates. It is also the most secure way, since the messages are sent from the server already signed and do not require a proxy server to apply any further processing. Signing official bulk email at message generation is the proposed method for NPS.

2. After Generation

The second alternative, implementing a proxy server after message generation, seems to be the easiest logically, but is far less secure than signing at message generation. This approach allows the emails to be generated at different systems, and signed at a single point. Applying a digital signature to a completed message is a relatively simple task, as long as the signer possesses valid certificates and corresponding private keys for the specific senders (BulkMail@nps.edu and SQLMgr@nps.edu). Each automated email generator we are targeting could be set to send unsigned messages to the signer; the signer would then forward it to the mail delivery agent. Essentially we would have a signing proxy server here that would effectively intercept

all mail from these two sources, apply a digital signature, and then send it to the users.

As previously stated, this ease of this method comes with serious vulnerabilities. The use of a proxy server adds another vector of attack for adversaries, and increases the complexity of both the system and the necessary security. To avoid this, the proxy must be configured to accept email only from the designated sources. The sole advantage of this append is that it does not affect production scripts other than changing the mail relay.

We created such a proxy server proof-of-concept in this fashion. It is not the most secure way to accomplish the goal of authenticated official bulk email, and should only be considered if signing at message generation is impossible.

3. Upon Receipt

The final alternative of signing mail when received is highly illogical as it describes a process of getting the message first then applying an authentication signature. This defeats the purpose since the user has already received the message!

THIS PAGE INTENTIONALLY LEFT BLANK

V. SIGNING MESSAGES WITH S/MIME

There are three standards-based approaches for signing mail today: S/MIME, PGP, and DomainKeys/DKIM. This chapter reviews S/MIME and discusses our results in signing S/MIME messages.

A. S/MIME

Multipurpose Internet Mail Extensions (MIME) is a communications standard that provides a common protocol for all email messages. This allows different operating systems with different mail programs to interpret email correctly. Secure MIME (S/MIME) is an extension to MIME that allows for digital signing and encryption.

S/MIME works through asymmetrical key algorithms (like RSA). A user must first obtain a certificate with a public/private key pair. It should also be signed from a trusted certificate authority, though users may create personal certificates and sign them with self-generated keys. S/MIME therefore authenticates the individual who signs the message, given trust in the individual or authority that signed the individual's certificate.

To create a digital signature in S/MIME, the user's email message is first encoded in MIME format. A digital hash is then created from the email. This hash is then encrypted with the sender's private key. Next, the mail program creates a new multipart MIME message, with the old message as the first part and the S/MIME signature consisting of the signed hash and the sender's certificate as the second part. This is the message that gets sent

across the Internet. On the recipient's side, the public key is extracted from the sender's certificate and used to decrypt the signature. The recipient then hashes the original message, and compares the result to the decrypted hash. If the two match, the receiver is assured that the message was not modified in transit and that the owner of the certificate sent it.

There are two usability problems with S/MIME. First, since S/MIME expands the original message by including the digital signature and attaching the sender's certificate, it requires a MUA with S/MIME support to verify the message correctly. Programs that do not implement S/MIME typically show the original message with an attachment; this file with .p7s attachment may confuse users who do not understand what it is. Second, many S/MIME agents will warn the user if a signature does not match, cannot be verified, or any other number of errors possibly associated with S/MIME.

One of the largest advantages to S/MIME is its industry acceptance. Most of the mainstream MUA'a implement the S/MIME standard, and can process these signed or encrypted messages.

B. IMPLEMENTING S/MIME

For this thesis, we decided to use a proxy SMTP server that would receive automatically generated emails from two different sources, apply a digital signature based on the source, then forward that message to a production server for delivery. We looked at three solutions for this proxy server: a product meant for extremely large enterprises, one that was designed for smaller institutions, and finally a

proxy server written in the Python language using a well-known library called Twisted.

1. ColdSpark Solutions

We reviewed the ColdSpark mail processing system. This company typically caters to Fortune 500 businesses, which need the ability to process millions of emails at one time; solutions start at $250,000.

2. PGP Universal

We tested the PGP Universal Server by OpenPGP. This product was more in line with the requirements and budget of our organization. We obtained from PGP an evaluation license for the PGP Universal server. PGP Universal Gateway Email Server is a product that performs a lot of functions, it acts as a router, is a stand-alone email server (complete with functionality and administrative rules regarding user mailboxes), and has a wide variety of options to implement rules based on different aspects of received or generated email. This product has a list price of $3,120. We configured the server running on a laptop with its own static IP and DNS entry on the NPS network, behind the firewall. We were able, with some difficulty, to import test certificates for both “BulkMail@nps.edu” and “SQLMgr@nps.edu.” These were both self-generated and signed certificates.

3. Python Scripted Server

Our third way of implementing a signing-proxy was to develop our own. We used a well-used set of libraries written in Python called Twisted to flesh out our SMTP server. Using the programming language Python allows us a very lightweight and easy to program application that is operating system independent. The Twisted framework is a networking engine, supporting numerous protocols, including SMTP [26]. The sample python script appears in Appendix A.

4. Obtaining Test Certificates

We obtained test certificates by going through a process instituted by the Defense Information Systems Agency (DISA). This process mimicked the actual process of obtaining email certificates within the Department of Defense, but all information we entered was fake test data (by instruction). The test certificates enabled us to digitally sign test messages through the use of OpenSSL via a python script.

Once the SMTP proxy server was running, it awaits an incoming message. When it receives an email, it will copy the headers so it can forward the signed message to the appropriate production SMTP server. The signing proxy then calls a function to sign the message. This function uses the OpenSSL command to create the signature. The OpenSSL command then returns the message + signature to the signing proxy. From there, the proxy forwards the signed message to a production SMTP server.

This proof of concept was a fairly trivial example, but it has serious security flaws that must be addressed before it is considered as a viable solution. A first vulnerability is the connection between the generating script and the proxy server. We created the test message via command line, but in a production system, we would use existing scripts and automation. The message is unsigned from this source to the proxy server, allowing an attacker the exact same attack vector that we are trying to solve. The adversary can spoof the bulk email “From” address, and send whatever phishing message to the proxy server. Without any verification, the server will sign the fraudulent email and forward it to the intended recipients. The recipients will see a valid digital signature and fall victim to the attack.

Another vulnerability of the proxy server is the storage of the certificates. The server needs to use the private keys for each address it is authorized to sign. Therefore the security of these keys is the same level as the security of the server itself. An adversary could compromise the server, obtain the certificates, and then apply a digital signature to any fraudulent email he/she wishes, undermining the value of digital authentication.

A third vulnerability with our test proxy is the openness that we allowed. The test proxy will accept any message with any From and To address. If this were to go into production, then an adversary has no roadblocks to spoofing any message they want. Clearly, additional access controls are required before this system is deployed.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. MAIL USER AGENTS AND S/MIME

This chapter consists of screenshots of different Mail User Agents (MUA) and what they show when there is a valid digital signature present in a message. There will be screenshots that show S/MIME and DomainKey/DKIM signatures.

A. S/MIME SIGNED BY COMMON ACCESS CARD VS THAWTE FREE EMAIL

This section shows what some different MUAs display when they look at a message digitally signed with S/MIME. It highlights a difference in how the DoD uses the fields within the personal certificate compared to an online signing authority (Thawte), and how this becomes an issue.

1. Microsoft Outlook

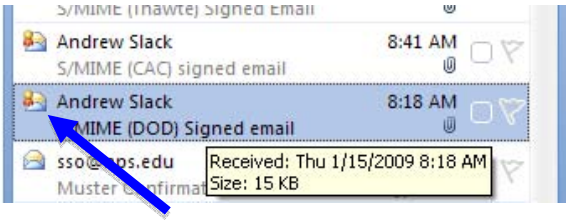


Figure 5. Microsoft Outlook Preview Screen

Figure 6. Microsoft Outlook Digital Signature Indicator

Microsoft Outlook contains an icon for a signed message from the preview screen, and a similar icon when the actual message is viewed. These indicate a valid signature, as shown in Figures 5 and 6.

Figure 7. Microsoft Outlook Digital Signature Details

By clicking on the icon, it will bring up a more detailed window, indicating that the signature is both valid and trusted.

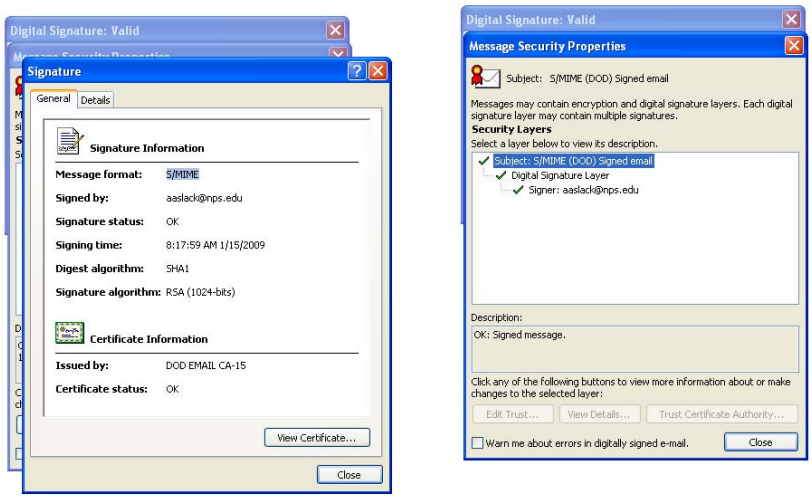


Figure 8. Microsoft Outlook S/MIME Details

Outlook can display more details about the signature, including who the signer is, and the identity of the certificate authority.

If we replace the DoD certificate with one that hasn't been checked against a revocation list, we highlight some security concerns within Outlook:

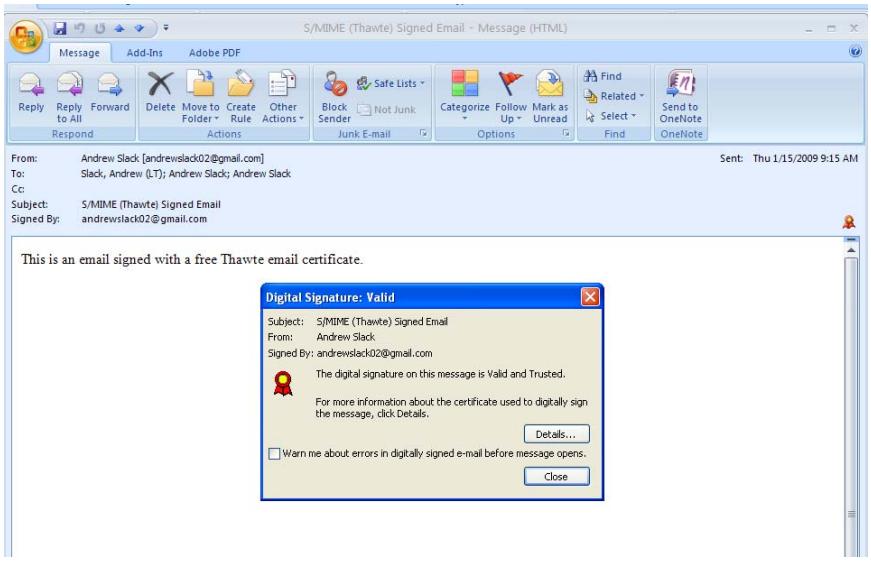


Figure 9. Microsoft Outlook Certificate Warning

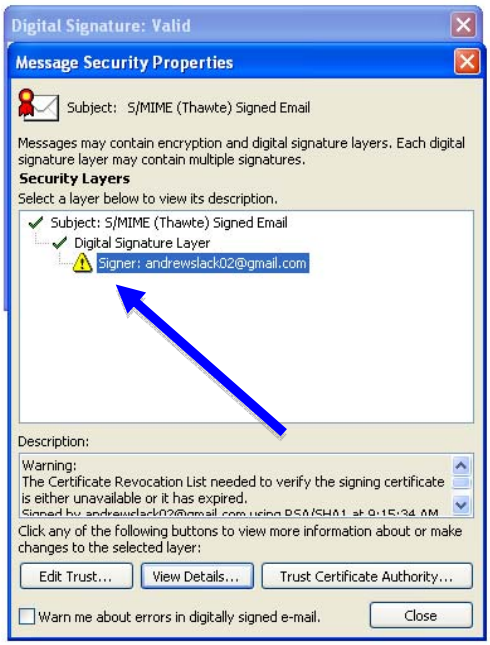


Figure 10. Microsoft Outlook Warning Properties

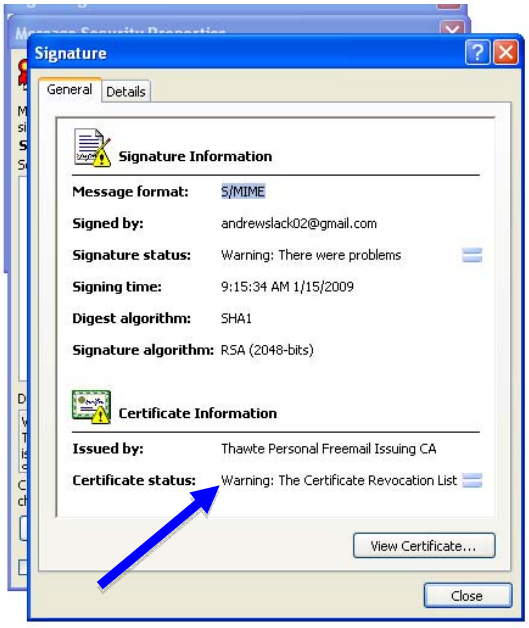


Figure 11. Microsoft Outlook Warning Details

Using a certificate from a free Internet certificate authority, such as Thawte, Microsoft Outlook accepts the

signature, and indicates that it is valid and trusted. Yet when details are shown, there is a yellow warning sign because Outlook cannot check the Certificate Revocation List for this certificate. Exploring the warning brings further details, but they are obfuscated within a poor layout of the window (Figure 11).

2. Apple Mail

This section will show what an accepted signature looks like in Apple Mail. In this case, I have manually accepted the certificate to show a valid signature. Without the trusting the DoD-issued certificate, Apple Mail will raise a flag, which is shown later in the chapter.

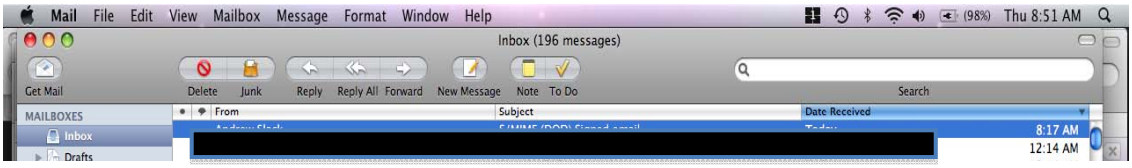


Figure 12. Apple Mail Signed Mail Preview

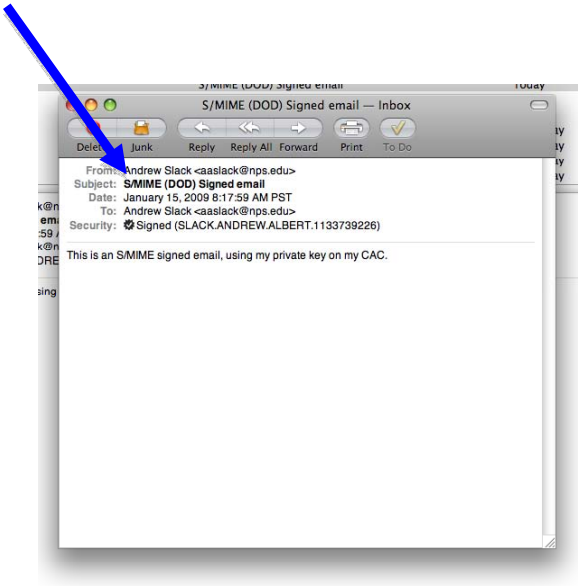


Figure 13. Apple Mail Signed Mail Indicator

Apple Mail is the MUA included with OS X. It does not show any indication of a signed message in the email preview list (Figure 12), but does show a security line within the actual message (Figure 13).

3. Microsoft Outlook Web Access

Microsoft Outlook Web Access is an HTTP based mail client. It shows a signature icon for mail that contains a digital signature. Inside the message though, it notifiesthe user that the message signature is not valid (Figure 15). Also, it doesn't show an attachment icon, but you can see the "<<smime.p7s>>" signature attachment below the message.

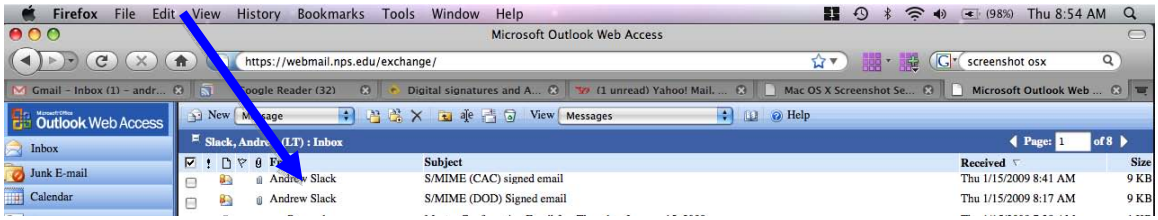


Figure 14. Microsoft Outlook Web Access Preview Screen

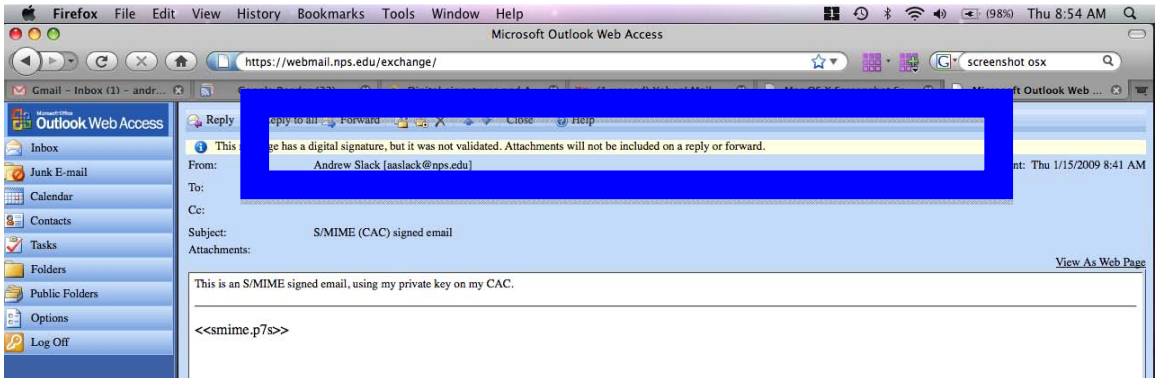


Figure 15. Microsoft Outlook Web Access Signature Warning

4. Entourage

Entourage is the mail transfer agent for Mac OS X included in the Microsoft Office suite. Here, it clearly shows an icon indicating a signed message in both the email list and preview panel. (Note that Entourage uses the wrong icon for a digital signature).

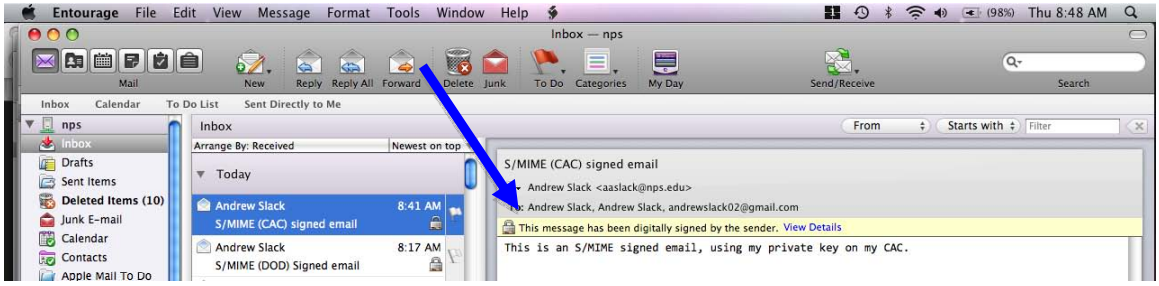


Figure 16. Microsoft Entourage Preview Panel

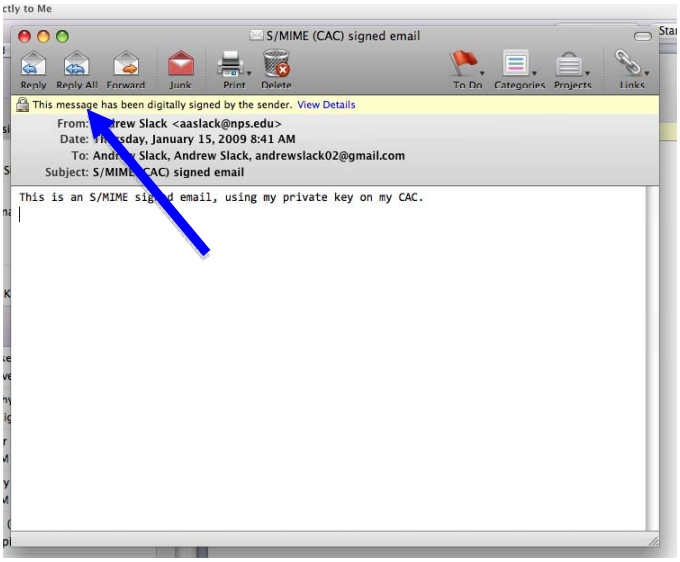


Figure 17. Microsoft Entourage Digital Signature Verification

By clicking on “view details,” Entourage shows a list of assurances from the digital signature. It is interesting that is has a green check next to the line that states that “Revocation information for this certificate has not been determined” (Figure 18).

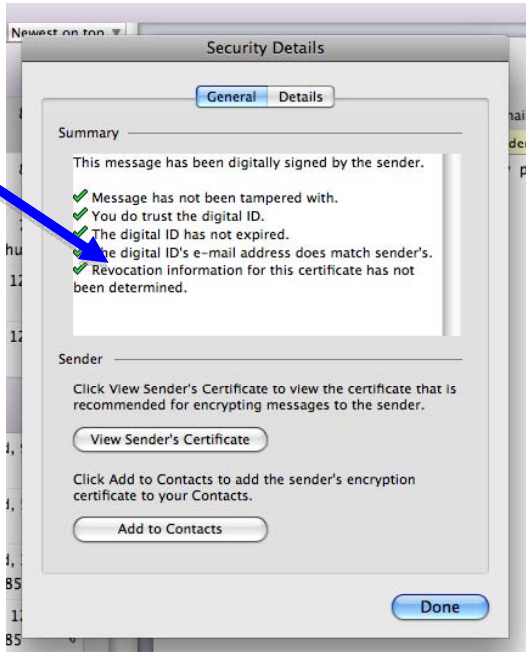


Figure 18. Microsoft Entourage Security Details

5. Thunderbird

Thunderbird is Mozilla’s open source Mail User Agent. It does not inherently trust the DoD certificate authority, so we can see the difference between a trusted and untrusted signature.

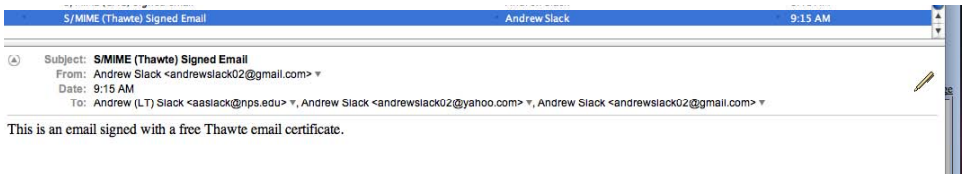


Figure 19. Mozilla Thunderbird Trusted Signature

Figure 20. Mozilla Thunderbird Signature Details

A valid signature will show an icon in the message itself (Figure 19). Clicking on the icon will bring up a summary of the signature (Figure 20).

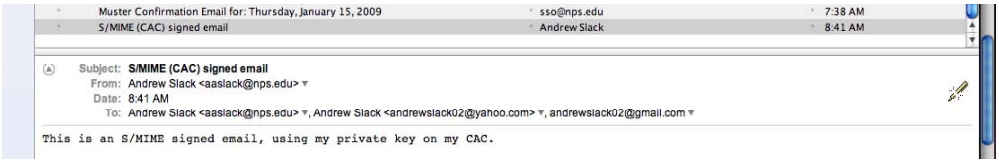


Figure 21. Thunderbird Unverified Signature

An invalid signature (or one that Thunderbird hasn't validated) is shown with a broken symbol (Figure 21). Clicking on the details shows a window that identifies the problem (Figure 22).

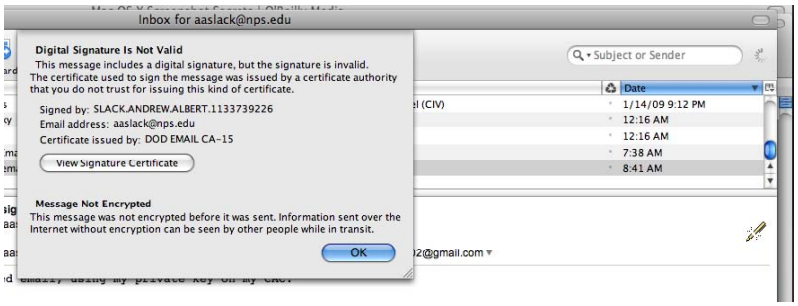


Figure 22. Thunderbird Unverified Signature Details

6. Google Mail (Gmail)

Most web-based browsers do not support S/MIME. Google's Gmail is no exception. Here we can see the signed message, but it has no icon or any indication that the message has a valid signature. It does (as all MUA's that do not support S/MIME) show the .p7s attachment.

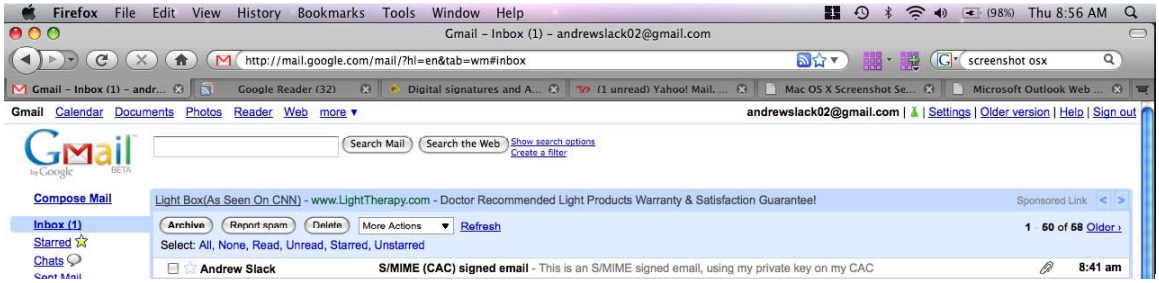


Figure 24. Google Mail Signed Message

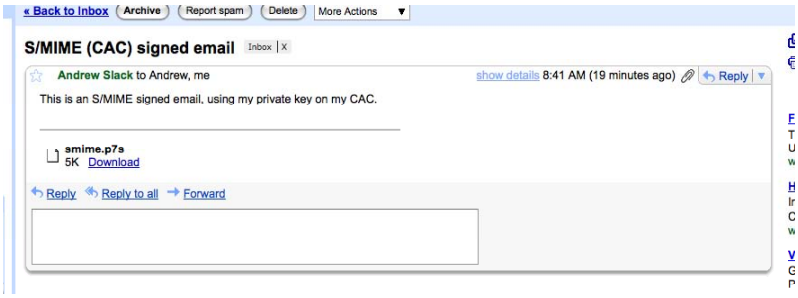


Figure 24. Google Mail Signed Message

The Firefox browser (by Mozilla) does have an available "S/MIME" plug-in for Gmail, but it only allows a user to decrypt, encrypt, or sign a message; not verify a signature (Figure 25).

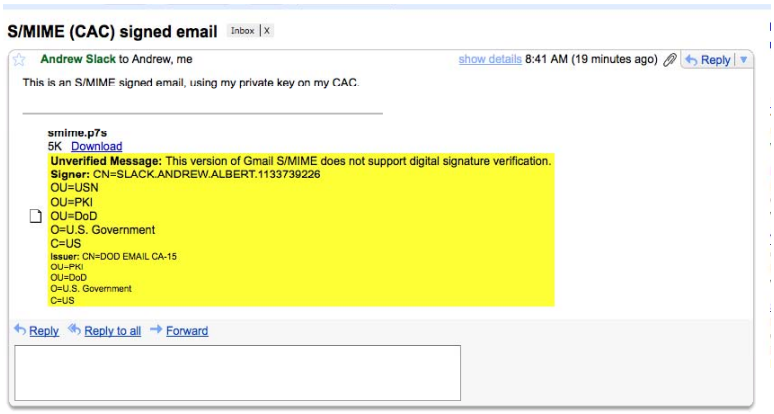


Figure 25. Firefox Plugin S/MIME Verification Error

7. Yahoo! Mail

Yahoo! Mail is another web-based mail user agent. It also does not support S/MIME, and shows the signature as an attachment. There is no S/MIME plug-in for Yahoo! Mail.

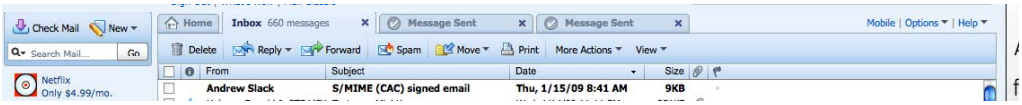


Figure 26. Yahoo! Mail Preview

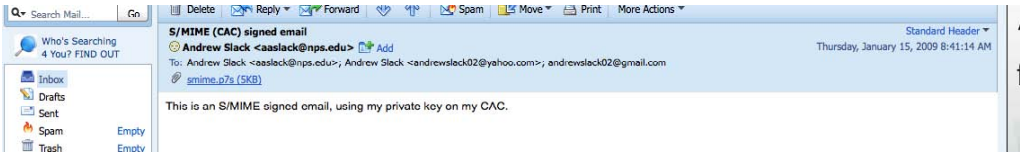


Figure 27. Yahoo! Mail Signed Message View

B. DOMAINKEY AND DKIM

Google's Gmail signs each of its messages with a DKIM signature to ensure that the message did indeed come from an "@gmail.com" address. Interestingly, Gmail also places a DomainKey signature in the header as well. Since both of these methods only affect optional header fields, MUA's that do not support DKIM or DomainKey simply ignore the signature, and treat the message as unsigned.

This is the full header that Google applies to the messages from Gmail. It includes both a DKIM and DomainKey signature.

```
From Andrew Slack Thu Jan 15 09:30:16 2009
Return-Path: <andrewslack02@gmail.com>
Authentication-Results: mta190.mail.re2.yahoo.com  from=gmail.com; domainkeys=pass (ok)
Received: from 209.85.218.20  (EHLO mail-bw0-f20.google.com) (209.85.218.20)
  by mta190.mail.re2.yahoo.com with SMTP; Thu, 15 Jan 2009 09:30:20 -0800
Received: by bwz13 with SMTP id 13so3348904bwz.17
        for <andrewslack02@yahoo.com>; Thu, 15 Jan 2009 09:30:16 -0800 (PST)
DKIM-Signature: v=1; a=rsa-sha256; c=relaxed/relaxed;
        d=gmail.com; s=gamma;
        h=domainkey-signature:received:received:message-id:date:from:to
         :subject:mime-version:content-type;
        bh=2qAR7pOiOFip6KP3V4yvG/6L2fL8KfsYRJYcq6lyHzw=;
        b=BUZ7kYDoYkBTDg/ttADhjHa+ZghljieU/OFz43rlaJY21kUHJyL0oPdQ4kELe9rrDb
         NFlGm/o0iAJsOPoLwU2jyZSkT4yjadKhzhDo+h+YmdPMH0Za9AJIEeAXR5uOctZY52R4
         yfHYbxnhOwBygd1KNu7LsKRqbRQE7+ahcK0hg=
DomainKey-Signature: a=rsa-sha1; c=nofws;
        d=gmail.com; s=gamma;
        h=message-id:date:from:to:subject:mime-version:content-type;
        b=aN85NCqaLZ/6DO7oDzSpNyZjP4GVnatNPdmHss27uD7rcENDE5TN/nkRQALw89vEUp
         FmQDBEvPCUavGoNS4psZyn/bFd5zuWGYMryw59Rzkq/cDtigWqdK+78qRxI2YNNynyF0
         EfuRZQCQ0I4iKssoe5KiFmByioDNH2ZGsrd+A=
Received: by 10.223.114.68 with SMTP id d4mr1911859faq.86.1232040616283;
        Thu, 15 Jan 2009 09:30:16 -0800 (PST)
Received: by 10.223.110.202 with HTTP; Thu, 15 Jan 2009 09:30:16 -0800 (PST)
Message-ID: <3ad02d360901150930s3d5f3982ie6e79ed90b436fef@mail.gmail.com>
Date: Thu, 15 Jan 2009 09:30:16 -0800
From: "Andrew Slack" <andrewslack02@gmail.com>
To: "Andrew (LT) Slack" <aaslack@nps.edu>,
          "Andrew Slack" <andrewslack02@yahoo.com>,
          "Andrew Slack" <andrewslack02@gmail.com>
Subject: This is a DKIM (Gmail) signed email
MIME-Version: 1.0
Content-Type: multipart/alternative;
          boundary="----=_Part_15066_23093755.1232040616277"
Content-Length: 531
```

1. Gmail

Gmail does not show that a message contains a DKIM or Domainkey signature, but if you expand the header, you can see the "signed-by" field.

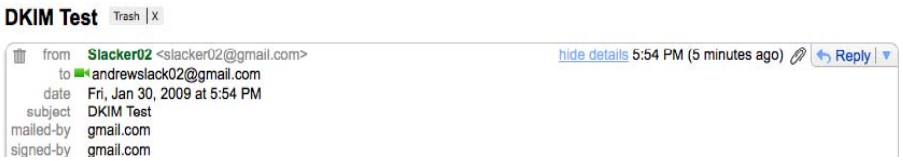


Figure 28. Google Mail DKIM/DomainKey Signature

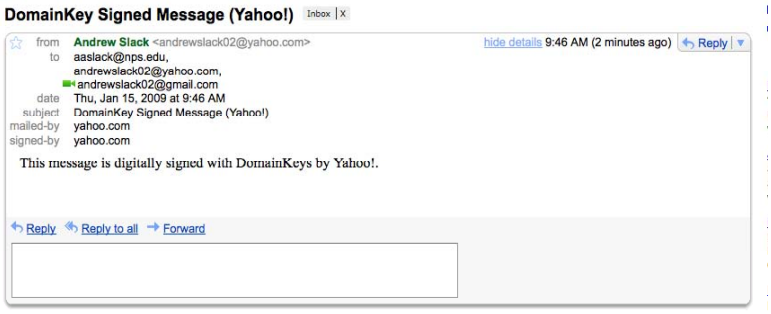


Figure 29. Google Mail DomainKey Signature

2. Yahoo!

Yahoo! will show an icon within the message, indicating that it contains a DomainKey signature.

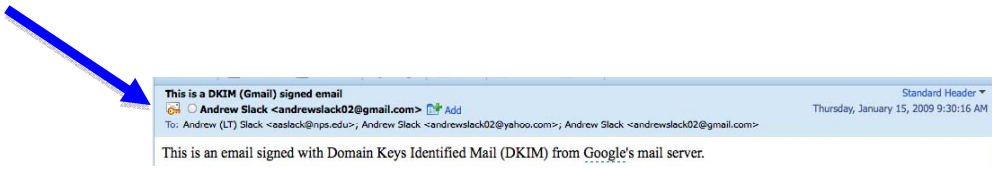


Figure 30. Yahoo! Mail DomainKey Signed Icon

Yahoo! ignores the DKIM signature, but verifies the DomainKey one.

3. Outlook, Thunderbird, Apple Mail, Entourage, Webmail

These MUA’s do not support DomainKeys or DKIM at this time. Even though the header of the message contains the DKIM or DomainKey signature, the MUA’s treat the message as if there is no signature present. This is one of the advantages of incrementally deploying DKIM and similar domain-level authentication technologies: Mail services that do not have the DKIM software installed will not display an error, or an unknown attachment; it will just treat the message as unsigned.

VII. CONCLUSION

Phishing, and more importantly Spear Phishing, attacks against DoD institutions will continue to grow in magnitude and sophistication as adversaries increase their understanding of both the intended target and the potential advantages of compromised information. These attacks are intended to defeat normal spam firewalls, masquerade as legitimate email within an organization, and target the human as a weak point in network security.

Official bulk email has the inherent attributes of authenticity and integrity, without the presence of a digital signature. The combination of spear phishing attacks with official bulk email creates an extremely dangerous vector of attack into Department of Defense networks. Attackers, who have already defeated the automated network defenses, now have added trust to their fraudulent email stolen from the spoofed automated bulk email address.

Digital signatures will help mitigate this vulnerability, and will assist in training individuals to recognize fraudulent emails despite the sophistication of the attack. We have shown that it is relatively simple to employ an S/MIME proxy server, with test certificates, into an operational enclave email system to apply a digital signature to auto generated email. This digital authentication solution will only be possible if organizations can move past the administrative roadblocks such as legacy email architecture, the fear of altering a

working system, and obtaining role-based certificates despite enabling policy already in place.

Department of Defense policy states the requirement for digital signatures on any email that meets certain criteria; criteria that is almost always contained in official bulk email. By not implementing a digital signature solution to automated official bulk email (whether it is an automated proxy server, or manual application), DoD enclaves are violating official policy.

The need for digital authentication on official bulk email is clear, the requirement for digital authentication on official bulk email is mandatory, and the automated solution is present.

LIST OF REFERENCES

[1] E. Allman, “E-mail authentication: what, why, how?” *ACM Queue.* Vol. 4, No. 9 (November 2006), pp. 30-34, DOI= http://doi.acm.org/10.1145/1180176.1180191, last accessed March 2009.

[2] P. Hallam-Baker, *dotCrime Manifesto: How to Stop Internet Crime*. Pearson Education, Inc. 2008.

[3] Department of Defense Instruction 8520.2, “Public Key Infrastructure (PKI) And Public Key (PK) Enabling.”

[4] Privacy Rights Clearinghouse Phishing Alert http://www.privacyrights.org/ar/phishing.htm, last accessed March 2009.

[5] J. W. Ragucci, S. A. Robila, "Societal Aspects of Phishing," *Technology and Society, 2006. ISTAS 2006, IEEE International Symposium on*, pp. 1-5, 8-10 June 2006, URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4375893&isnumber=4375875, last accessed March 2009

[6] J. Kirk, “PayPal asks ISPs to block unsigned email,” IDG News Service, 27 March 2007.

[7] S. A. Robila and J. W. Ragucci, “Don't be a phish: steps in user education,” in *Proceedings of the 11th Annual SIGCSE Conference on innovation and Technology in Computer Science Education (Bologna, Italy, June 26 - 28, 2006), ITICSE '06*. ACM, New York, NY, pp.237-241. DOI= http://doi.acm.org/10.1145/1140124.1140187, last accessed March 2009.

[8] A.J. Ferguson, “Fostering E-mail Security Awareness: The West Point Carronade,” *Educause Quarterly*. Vol. 28, No. 1. 2005, pp. 54-57.

[9] Anti-Phishing Working Group, “Phishing Activities Trends Report Q1/2008,“ http://www.antiphishing.org/reports/apwg_report_Q1_2008.pdf, last accessed March 2009.

[10] B. Brewin, “DoD Battles Spear Phishing,” *Federal Computer Weekly*, http://archive.cert.uni-stuttgart.de/isn/2006/12/msg00114.html, last accessed March 2009.

[11] J. B. Postel, “RFC 821: Simple Mail Transfer Protocol,” Information Sciences Institute, University of Southern California. August 1982, http://www.ietf.org/rfc/rfc0821.txt, last accessed March 2009.

[12] J. Klensin, “RFC 2821: Simple Mail Transfer Protocol,” The Internet Society, 2001. http://www.ietf.org/rfc/rfc2821.txt, last accessed March 2009.

[13] J. Klensin, “RFC 5321: Simple Mail Transfer Protocol.” Network Working Group. October 2008, http://www.rfc-editor.org/rfc/rfc5321.txt, last accessed March 2009.

[14] Alma Whitten and J. D. Tygar, "Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0," in *Proceedings of the 8th USENIX Security Symposium*, August 1999. http://portal.acm.org/citation.cfm?id=1251435, last accessed March 2009.

[15] L. F. Cranor and S. L. Garfinkel, *Security and Usability: Designing Secure Systems That People Can Use*. O’Reilly Media Inc. 2005. p. 247.

[16] S. L. Garfinkel, D. Margrave, J. I. Schiller, E. Nordlander, and R. C. Miller, “How to make secure email easier to use” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Portland, Oregon, USA, April 02 - 07, 2005). *CHI '05*. *ACM*, New York, NY, pp. 701-710.

[17] S. L. Garfinkel and R. C. Miller, “Johnny 2: a user test of key continuity management with S/MIME and Outlook Express,” in *Proceedings of the 2005 Symposium on Usable Privacy and Security* (Pittsburgh, Pennsylvania, July 06 - 08, 2005). *SOUPS '05*, vol. 93. *ACM*, New York, NY, pp. 13-24, DOI= http://doi.acm.org/10.1145/1073001.1073003, last accessed March 2009.

[18] S. Gaw, E.W. Felten, and P. Fernandez-Kelly, “Secrecy, flagging, and paranoia: adoption criteria in encrypted email,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Montréal, Québec, Canada, April 22 - 27, 2006).

[19] G. D. Fritsche and S. K. Rodgers, “Encryption technologies: testing and identifying campus needs,” in *Proceedings of the 35th Annual ACM SIGUCCS Conference on User Services* (Orlando, Florida, USA, October 07 - 10, 2007). *SIGUCCS '07*. ACM, New York, NY, pp.109-112. DOI= http://doi.acm.org/10.1145/1294046.1294071, last accessed March 2009.

[20] J. Linn, “RFC 1421: Privacy Enhancement for Internet Electronic Mail: Part I: Message Encryption and Authentication Procedures,” Network Working Group, February 1993, http://tools.ietf.org/rfc/rfc1421.txt, last accessed March 2009.

[21] Philip R. Zimmermann, “Why I Wrote PGP”. Part of the Original 1991 PGP User's Guide (updated in 1999), http://www.philzimmermann.com/EN/essays/WhyIWrotePGP.html, last accessed March 2009.

[22] J. Goodman, G. V. Cormack, and D. Heckerman, “Spam and the ongoing battle for the inbox,” *Commun. ACM.* Vol. 50, No. 2 (February. 2007), pp. 24-33, DOI= http://doi.acm.org/10.1145/1216016.1216017, last accessed March 2009.

[23] JTF-GNO Fact Sheet, http://www.stratcom.mil/factsheets/gno/, last accessed March 2009.

[24] Department of Defense NAVADMIN 248/08 “Implementation of Navy Electronic Mail (Email) Digital Signature Policy.”

[25] Department of the Navy CIO Message DTG 202041Z AUG 07, “Department of the Navy Security Guidance For Personal Electronic Devices (PED).”

[26] Twisted Matrix Labs, http://twistedmatrix.com/trac/wiki/TwistedProject, last accessed March 2009.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX

A. PYTHON CODE - SIGNING PROXY SERVER

```
#
# Original Copyright (c) 2001-2004 Twisted Matrix Laboratories.
# http://twistedmatrix.com/projects/mail/documentation/examples/
# See LICENSE for details.
#
# You can run this module directly with:
#    twistd -ny emailserver.tac


"""
A signing email proxy.
"""

myname  = "Signing Mail Proxy v0.0.1"
myproxy = "virginia.nps.edu"
pemfile = "BulkMail.pem"

from zope.interface import implements
from twisted.internet import defer
from twisted.mail import smtp
import smime

# http://python.net/crew/mwh/apidocs/twisted.mail.smtp.IMessageDelivery.html
class ConsoleMessageDelivery:
    implements(smtp.IMessageDelivery)

    def __init__(self):
        self.toaddrs = []

    def receivedHeader(self, helo, origin, recipients):
        self.helo = helo
        return "Received: "+myname

    def validateTo(self, user):
        # Right now, accept all messages. Eventually we want to only accept To addresses
        # that the destination will accept.
        self.toaddrs.append(user)              # make a copy
        return lambda: ConsoleMessage(self)

    def validateFrom(self, helo, origin):
        # All addresses are accepted
        self.fromaddr = origin
        return origin


class ConsoleMessage:
    implements(smtp.IMessage)

    def __init__(self,delivery):
        self.lines = []
        self.delivery = delivery

    def lineReceived(self, line):
        self.lines.append(line)

    def connectionLost(self):
        # There was an error, throw away the stored lines
        self.lines = None

    def eomReceived(self):
```

```
        import email
        msg = "\r\n".join(self.lines))
        msg = smime.sign(msg,pemfile)

       # Send out the message using smtplib
       import smtplib
       server = smtplib.SMTP(myproxy)
       server.sendmail(self.delivery.fromaddr,self.delivery.toaddrs,msg)
       server.quit()
        return defer.succeed(None)

class ConsoleSMTPFactory(smtp.SMTPFactory):
    def __init__(self, *a, **kw):
        smtp.SMTPFactory.__init__(self, *a, **kw)
        self.delivery = ConsoleMessageDelivery()

    def buildProtocol(self, addr):
        p = smtp.SMTPFactory.buildProtocol(self, addr)
        p.delivery = self.delivery
        return p

def main():
    from twisted.application import internet
    from twisted.application import service

    a = service.Application("Console SMTP Server")
    internet.TCPServer(2500, ConsoleSMTPFactory()).setServiceParent(a)

    return a

application = main()
```

B. PYTHON CODE – OPENSSL

```
#!/usr/bin/python
#
# sign a mail message using openssl

def sign(msg,keyfile):
    """Sign the RFC822 message using openssl.
    MSG should be a string.
    Returns a string.
    """
    from subprocess import Popen,PIPE

    # Get the specific headers we care about to carry through
    msg822 = email.message_from_string(msg)
    headers = ""
    for field in ['To','From','Subject','Message-Id','x-mailer']:
        val = msg822.get(field)
        if val: headers = headers + field + ": " + val + "\r\n"


    # Sign the message

    proc = Popen(['openssl','smime','-sign','-signer',keyfile,'-inkey',keyfile],
                 stdin=PIPE,stdout=PIPE,stderr=PIPE)

    (signed_message,stderr) = proc.communicate(msg)

    # Add back those headers
    signed_message = headers + signed_message

    if stderr:
        raise ValueError,stderr
```

```
    if proc.returncode:
        raise ValueError,"OpenSSL returned %d" % proc.returncode
    return signed_message


if __name__=="__main__":
    import sys,email
    msg = sys.stdin.read()
    signed_message = sign(msg,sys.argv[1])
    if len(sys.argv)>1:
        import smtplib
        server = smtplib.SMTP("mx1.balanced.spunky.mail.dreamhost.com",25)
        #server.set_debuglevel(1)
        server.sendmail("BulkMail@nps.edu",["test_recipient@nps.edu"],signed_message)
        server.quit()
```

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California